Czech ACM Student Chapter
CTU in Prague, Dept. of Computer Science and Engineering
VŠB TU Ostrava, FEI, Dept. of Computer Science
SUT Bratislava, FEI, Dept. of Computer Science and Engineering

CTU Open Contest 2002

Formatting Text

`format.p`, `format.c`, `format.C`

Writing e-mails is fun, but, unfortunately, they often do not look very nice, mainly because not all lines have the same lengths. Summit representatives want to sent nicely formatted e-mails and your task is to write an e-mail formatting program for them.

The easiest way to perform this task would be to insert more spaces between the words in lines which are too short. But this is not the best way. Consider the following example:

```
****************************
This is the example you are
actually considering.
```

Let us assume that we want to get lines as long as the row of stars. Then, by simply inserting spaces, we would get

```
****************************
This is the example you  are
actually        considering.
```

But this looks rather odd because of the large gap in the second line. By moving the word "`are`" from the first to the second line, we get a better result:

```
****************************
This  is  the  example   you
are  actually   considering.
```

Of course, this has to be formalized. To do this, we assign a badness to each gap between words. The badness assigned to a gap of n spaces is $(n-1)^2$. The goal of the program is to minimize the sum of all badnesses. For example, the badness of the first example is $1+7^2=50$, whereas the badness of the second one is only $1+1+1+4+1+4=12$.

In the output, every line should start and end with a word. I.e., there cannot be a gap at the beginning or at the end of a line. The only exception to this are the lines that contain a single word only. You can output such lines provided that the word is put at the beginning of the line. A badness of 500 is assigned to such a line if it is shorter than it should be. Of course, in this case, the length of the line is simply the length of the single word.

Input Specification

The input file contains a text consisting of several paragraphs. Each paragraph is preceded by a line containing a single integer N, the desired width of the paragraph, $1 \leq N \leq 80$. Paragraphs consist of zero or more lines which contain one or more words each. Words consist of characters with ASCII codes between 33 and 126, inclusive, and are separated by spaces (possibly more than one). No word will be longer than the desired width of the paragraph. The total length of all words of one paragraph will not be more than 10000 characters. No line will be longer than 100 characters.

Each paragraph is terminated by exactly one blank line. There is no limit on the number of paragraphs in the input file. The input file will be terminated by a paragraph description starting with $N = 0$. This paragraph should not be processed.

Output Specification

For each paragraph, find the formatting with the lowest possible badness and output the sentence "`Minimal badness is` B`.`" where B is the badness of the best possible formatting.

Sample Input

```
28
This   is   the   example   you   are
actually considering.

25
Writing e-mails is fun, and with this program,
they even look nice.

0
```

Output for Sample Input

```
Minimal badness is 12.
Minimal badness is 14.
```